

PLP - Recuperatorio del Segundo Parcial - 1^{er} cuatrimestre de 2023

Este examen se aprueba obteniendo al menos dos ejercicios bien menos (B-) y uno regular (R). Las notas para cada ejercicio son: -, I, R, B-, B. Entregar cada ejercicio en hojas separadas. Poner nombre, apellido y número de orden en todas las hojas, y numerarlas. Se puede utilizar todo lo definido en las prácticas y todo lo que se dio en clase, colocando referencias claras. El orden de los ejercicios es arbitrario. Recomendamos leer el parcial completo antes de empezar a resolverlo.

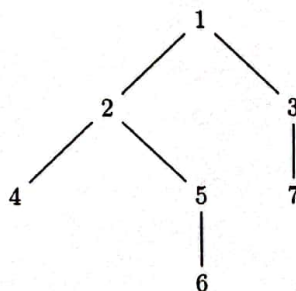
Ejercicio 1 - Programación Lógica

Implementar los predicados respetando en cada caso la instanciación pedida. Los generadores deben cubrir todas las instancias válidas de aquello que generan sin repetir dos veces la misma. No usar cut (!) ni predicados de alto orden como `setof`, con la única excepción de `not`. No está permitido utilizar estructuras auxiliares.

Trabajaremos con árboles binarios, usando `nil` y `bin(I, V, D)` para representarlos en Prolog. Como ejemplo, usaremos el siguiente hecho:

```
arbol(bin(bin(bin(nil,4,nil),2,bin(nil,5,bin(nil,6,nil))),1,bin(bin(nil,7,nil),3,nil))).
```

que representa el árbol:



- a) Definir el predicado `camino(+A,-C)` que es verdadero cuando `C` es un camino desde la raíz del árbol `A` hasta alguna de sus hojas. Por ejemplo:

```
?- arbol(A), camino(A,C).
C = [1,2,4];
C = [1,2,5,6];
C = [1,3,7];
false.
```

- b) Definir el predicado `caminoMasLargo(+A,-C)` que es verdadero cuando `C` es el camino más largo de `A`. Por ejemplo:

```
?- arbol(A), caminoMasLargo(A,C).
C = [1,2,5,6];
false.
```

- c) Definir el predicado `caminoUnicoDeLong(+A,+N,-C)` que es verdadero cuando `C` es el único camino de `A` que tiene longitud `N`. Por ejemplo:

```
?- arbol(A), caminoUnicoDeLong(A,4,C).
C = [1,2,5,6];
false.
?- arbol(A), caminoUnicoDeLong(A,3,C).
false.
```

Ejercicio 2 - Objetos

a) Sea el siguiente código escrito en JavaScript:

```
function Slime(energía) { this.energía = energía; }

Slime.prototype.actualizarEnergía = function(valor){ this.energía += valor; }

function SlimeSanador(energía, poder) {
  Slime.call(this, energía);
  this.poder = poder;
  this.curar = function(paciente) {
    paciente.actualizarEnergía(this.poder);
  }
}

let a = new SlimeSanador(10, 40);

Object.setPrototypeOf(SlimeSanador.prototype, Slime.prototype);

Slime.prototype.reproducirse = function() {
  if (this.energía >= 10) {
    let cría = Object.create(this);
    cría.energía = 10;
    this.actualizarEnergía(-10);
    return cría;
  }
}

let b = new Slime(10);
a.curar(b);
let c = b.reproducirse();
b.comidaFavorita = "gelatina"
c.películaFavorita = "los cazafantasmas"
```

Indicar qué mensajes (*energía*, *actualizarEnergía*, *poder*, *curar*, *reproducirse*, *comidaFavorita* o *películaFavorita*) pueden responder los objetos a, b y c al finalizar la ejecución de la última línea, en qué objeto/s está definido cada mensaje; y describir las cadenas de prototipos.

b) Sean las siguientes definiciones en cálculo Sigma:

$$\text{error} \stackrel{\text{def}}{=} []$$
$$\begin{aligned} \text{Lista} \stackrel{\text{def}}{=} [& \text{new} = \zeta(z) [\text{isEmpty} = \zeta(s) z.\text{isEmpty}(s), \\ & \text{cons} = \zeta(s) z.\text{cons}(s), \\ & \text{head} = \zeta(s) z.\text{head}(s), \\ & \text{tail} = \zeta(s) z.\text{tail}(s)], \\ & \text{isEmpty} = \lambda(s) \text{True}, \\ & \text{cons} = \lambda(s) \lambda(x) ((s.\text{isEmpty} := \text{False}).\text{head} := x).\text{tail} := s, \\ & \text{head} = \lambda(s) \text{error}, \\ & \text{tail} = \lambda(s) s] \end{aligned}$$

Considerar definidos los objetos *true*, *false* y *cero* vistos en clase. Definir la función *longitud* que calcule la longitud de una lista. Por ejemplo, tiene que valer

$$\text{longitud}(\text{Lista}.\text{new}.\text{cons}(\text{cero}).\text{cons}(\text{cero})) \longrightarrow \text{dos'}$$

Ejercicio 3 - Resolución

Considerar las siguientes definiciones en Prolog:

```
frutal(frutilla).  
frutal(banana).  
cremoso(banana).  
cremoso(americana).  
cremoso(frutilla).  
leGusta(X) :- frutal(X), cremoso(X).  
cucurucho(X,Y) :- leGusta(X), leGusta(Y).
```

y la consulta:

?- cucurucho(X,Y).

- a) Convertir la base de conocimientos y la consulta a forma clausal.
- b) Utilizar el método de resolución para obtener el resultado de la consulta.
- c) La resolución utilizada en el inciso anterior, ¿fue SLD? Justificar. En caso de ser SLD, ¿respeto el orden en que Prolog hubiera resuelto la consulta? ¿Por qué?